Datorteknik DT509G HT20 Home Exam

26.08.2021

- This a take-home exam. This means that you are allowed to use the book and search for reference material on the internet.
- If you choose to use help materials, you **need to reference them**. Not referencing a source will result in an immediate fail of the whole exam.
- All text needs to be written directly by you. Quoting or copying formulations from another source (even if a reference is provided) will be considered as cheating.
- You are not allowed to use group discussions to solve the exercises. Suspected cases of copying from another student will result in an immediate fail of the whole exam and referal to the academic integrity board for all students involved.
- This exam has been personally created for you, so provide answers only to the questions you receive here.
- You can use any typesetting software (word, open office, google docs, latex) or write by hand and scan the paper. The final submission should be in the form of 1 single PDF file.
- You have 4 hours to solve the exam. Late submissions will not be accepted.
- The exam is 50 points, 25 points are needed to pass.
- **Before conitnuing with the exam you must read the next page with official instructions by the school**.

Instruktioner inför digital hemtentamen/ examination

Jag vill undvika fusk – hur gör jag? / All form av fusk anmäls

- **Du ska följa instruktionerna för uppgiften.** Om du är osäker, fråga ansvarig lärare om något i instruktionerna är oklart.
- **Du får inte samarbeta.** Det här är en individuell examinationsuppgift. Du ska inte prata med någon, ställa frågor till eller ta hjälp av andra studenter eller kurskamrater. Att hjälpa andra under en individuell examination är också fusk.
- **Lägg undan mobilen.** Stäng av sociala medier.
- **Du *kan* använda hjälpmedel.** Det är tillåtet att använda alla form av skriftlig hjälpmedel. Alla hjälpmedel muste refereras.
- **Du får inte använda andra formuleringar än dina egna.** Dina svar ska vara självständigt formulerade och redovisa dina kunskaper. Det betyder att inga citat ska förekomma i dina svar.
- **Dina svar kontrolleras via Urkund.**
- **All misstanke om fusk anmäls** till universitetets rektor och kan leda till en prövning i universitets disciplinnämnd.

Konsekvenser av fusk

Om du fuskar kan detta leda till en avstängning som kan få följder både för dina studier och privat:

- Uteblivet studiemedel som t.ex. som kan påverka din möjlighet att behålla din bostad.
- Ingen åtkomst till digitala plattformar.
- Du kan behöva meddela dina kursare om att du är fälld för fusk, om du t.ex. ingår i ett grupparbete på en pågående kurs men blir avstängd.
- Tillfällen för examination går förlorade vilket kan innebära att du inte kommer vidare i dina studier nästa läsperiod/termin och din studiegång blir därmed påverkad.
- Beslutet är en offentlig handling som begärs regelbundet ut av en nyhetsbyrå. Och kan även begäras ut av framtida arbetsgivare eller andra.
- Om du fuskar dig igenom din utbildning har du inte den kunskap som arbetsmarknaden förväntar av dig.

OBS: **Tänk efter en gång till innan du påbörjar och genomför din tentamen!**

1 Architectures, Digital Logic and Representation (10 points)

a: Explain what is the binary coded decimal representation and why would you want to use that instead of a floating point representation? (2 points)

b: A 16 bit unsigned floating-point representation uses 8 bits to represent the exponent. The exponent itself is represented as a two's complement signed integer. Assuming that the mantissa is normalized, what is the smallest negative number you can represent in this data type? Explain how the representation works and provide calculations. Convert the number 3.141516 to a float in this representation. (4 points)

c: Parallelism is a general paradigm that is used throughout modern computer architectures. Give one example of a use of parallelism within each of the following: the IO BUS, and memory. (2 points)

d: What is the difference between a half-adder and a full-adder circuit? When would you use each of theem? (2 points)

2 Processors (20 points)

In this exercise we will use a 32-bit processor with 64 general-purpose registers and a dedicated instruction pointer register (*ipp*). The word-size is 32 bits and the processor has separate instruction and data memories. The processor implements the following instructions:

add Adds the integers from registers r_a and r_b, plus an immediate value offset and stores the result in r_a.

sub Computes $r_a = r_a - r_b + offset$.

load Load an integer from data memory into register r_b. The integer is located at data address $r_a + offset$.

imd Load an imediate value from $offset$ into register r_a.

store Store an integer from register r_b into the data memory at location $r_a + offset$.

jump Set the program counter *ipp* to the integer address at location $r_a + offset$ in instruction memory.

ble Jumps to location $ipp + offset$, if $r_a <= r_b$.

noop No operation, idle.

a: Construct a binary representation for each instruction. Describe if you are using a fixed operand number, and if yes, how many operands. Try to optimize your encoding for achieving the largest possible number represented as an offset. Assuming that you are using a two's complement representation for the offset, what is the maximum number of instructions we can jump over in a single call to `jump`? (3 points)

b: A colleague of yours suggests to extend the architecture with register banks. Is that a good idea and why? Describe under what conditions would the use of register banks be beneficial. (2 points)

The architecture has now been extended with the following 3-stage pipeline:

- Stage (1): the instruction is parsed and operands are fetched from registers.
- Stage (2): ALU operations are carried out
- Stage (3): Results are stored in destination registers, memory is interfaced for read-/write operations.

c: Given the following instruction: `load r14(0x2A),r2`, where the value in brackets represents an offset. Explain how that instruction would be processed by each stage of the pipeline. (2 point)

d: Using the instruction set above, write a program block that calculates the value of $k! = 1 \times 2 \times 3 \cdots \times k$. Assume that k is an argument that has been provided to your block and is stored in `r0`. Hint: implement a subroutine that computes the multiplication operation `mul ra,rb` first. (4 points)

Given the following assembly code snippet:

```
         imd    r0(0x01)        # set immediate value
         imd    r1(0x04)        # set immediate value
         imd    r2(0x00)        # set immediate value
         imd    r3(0x0F)        # set immediate value
         load   r3(0x0A),r4     # load data in r4
label1:  ble    r2,r1,label2    # branch if r1<=r2
         add    r0,r4           # r0=r0+r4
         add    r2,r0           # r2=r0+r2
         jmp    label1          # jump to label1
label2:  store  r3,r4           # memory[r3]=r4
```

e: How many pipeline stalls will occur over the execution of the whole program? Is there a way to re-arrange the instructions to reduce the number of stalls and how? How would that change if the pipeline is modified to forward results after stage (2), and what does it mean to forward results? (4 points) f: Explain how a two stage assembler would process the code above to produce a binary executable program. Illustrate what the binary would look like after each stage pass and roughly translate the assembly code to machine instructions in your chosen encoding scheme. "Execute" the program by following through each instruction and show the value of all relevant registers at each cycle of execution. (5 points)

3 Memory (10 points)

a: A 32-bit computer architecture implements demand paging to provide a virtual memory space for user-space processes. The system is equipped with 4 GB of physical memory, which is symmetrically distributed on four physical memory banks. An MMU interfaces the four physical controllers. Assume that the operating system blocks just 512 MB of space and the remaining memory should be used for the demand paging system. Explain how demand paging works. Assuming that we want to use 2048 frames, how much memory would each page hold? Assume that the page table is stored within the region of memory reserved by the operating system. Explain what would be the trde-off between storing the page table in memory and on the MMU. (4 points)

b: A Direct Mapped Memory Cache is implemented on a 64-bit architecture. The cache can hold 512 blocks of memory, each of which is 1024 bytes long. How many bits are necessary in order to represent the tag of each of these blocks? Explain how the cache works and how powers of two are used to parse a byte address and find the correct data stored in the cache. Provide an illustration of how address translation works. (4 points)

c: A system implements a single-level cache between a processor and a memory unit. Under what conditions would it be beneficial to add one more cache level and why? (2 points)

4 I/O architecture (10 points)

a: How does the fetch-execute cycle need to be modified in order to implement interrupt-driven I/O? (2 points)

b: Explain when and under what conditions is I/O buffering beneficial? (3 points)

c: Explain the difference between interrupt-driven and polling I/O. Give an example of each. What are the demands on the software and hardware side in order to use interrupt-driven I/O? (3 points)

d: What is a bridge in the context of an I/O Bus and what function does it fulfill? (2 points)